

(19) World Intellectual Property Organization  
International Bureau



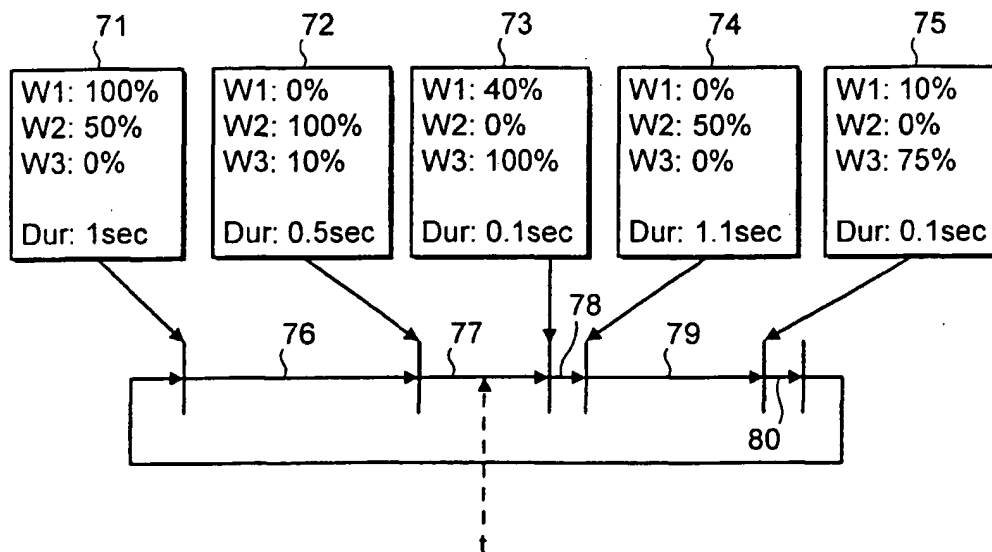
(43) International Publication Date  
15 November 2001 (15.11.2001)

PCT

(10) International Publication Number  
**WO 01/86629 A2**

- (51) International Patent Classification<sup>7</sup>: **G10H**
- (21) International Application Number: **PCT/GB01/01991**
- (22) International Filing Date: **4 May 2001 (04.05.2001)**
- (25) Filing Language: **English**
- (26) Publication Language: **English**
- (30) Priority Data:
- |           |                                |    |
|-----------|--------------------------------|----|
| 0010967.8 | 5 May 2000 (05.05.2000)        | GB |
| 0010969.4 | 5 May 2000 (05.05.2000)        | GB |
| 0011178.1 | 9 May 2000 (09.05.2000)        | GB |
| 0022164.8 | 11 September 2000 (11.09.2000) | GB |
| 0030979.9 | 19 December 2000 (19.12.2000)  | GB |
- (71) Applicant (for all designated States except US): **SSEYO LIMITED** [GB/GB]; Highview House, Charles Square, Bracknell, Berkshire RG12 1DF (GB).
- (72) Inventors; and
- (75) Inventors/Applicants (for US only): **COLE, John, Tim** [GB/GB]; Sseyo Limited, Highview House, Charles Square, Bracknell, Berkshire RG12 1DF (GB). **LEACH, Jeremy, Louis** [GB/GB]; Sseyo Limited, Highview House, Charles Square, Bracknell, Berkshire RG12 1DF (GB).
- (81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.
- (84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).
- Published:  
— without international search report and to be republished upon receipt of that report
- For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: **AUTOMATED GENERATION OF SOUND SEQUENCES**



(57) Abstract: A method of generating a sound sequence includes representing a variety of waveforms to be combined by a plurality of sample values, and then mixing the waveforms by combining their sample values in proportions that vary during a time interval (76 to 80). At the end of any time interval when the contribution of a particular waveform to the mix is zero, that waveform may be replaced by another waveform, and the process continued.

### Automated Generation of Sound Sequences

This invention relates to methods and systems for automated generation of sound sequences, and especially (though not exclusively) of sound sequences in the form of music.

The automated creation of music has a long history, going back at least as far as Mozart's use of musical dice. One of the first musical works generated by a computer was L. Hiller's Illiac suite. Since that time, of course, the sophistication of computer-generated music or more generally audio sequences has increased substantially.

Systems for creating musical sequences by computer may conveniently be divided up into two areas, which have been called "non-generative" and "generative". Non-generative systems include deterministic systems which will produce the same sequences every time, along with systems that simply replay (perhaps in a random or other order) pre-composed sections of music.

The vast majority of current systems which produce musical output make use of this type of approach, for example by selecting and playing a particular predefined sequence of notes at random when the key is pressed or a mouse button clicked. Generative Music Systems, on the other hand, may be considerably more complex. Such systems generate musical content, typically note by note, on the basis of a higher-level of musical knowledge. Such systems either explicitly or implicitly are aware of a variety of musical rules which are used to control or influence the generation of the music. In some systems, the rules may operate purely on the individual notes being generated, without imposing any form of higher order musical structure on the output; in such systems, any musical order that arises will be of an emergent nature. More sophisticated systems may include higher-level rules which can influence the overall musical structure. Generative Music Systems will normally create

musical content "on the fly", in other words the musical sequences are built up note by note and phrase by phrase, starting at the beginning and finishing at the end. This means that – in contrast with some of the non-generative systems - the musical content can be generated and played in real time: there is no need  
5 for example for the whole of the phrase to be generated before the first few notes of the phrase can be played.

For our present purposes, the essential features of a generative music system are that it generates musical content in a non-deterministic way, based upon a  
10 plurality of musical rules (which may either be implicit within the software or which may be explicitly specified by either the program writer or the user of the program). By analogy, a generative sound system produces non-deterministic sound sequences based upon sound-generation rules.

15 According to a first aspect of the present invention there is provided a method of generating a sound sequence comprising representing first, second and third waveforms by a plurality of sample values; during a first time interval, mixing the first and second waveforms by combining their respective samples in proportions that vary during the first time interval; at the end of the first  
20 interval, when the proportion of the first waveform is zero, replacing the first waveform with the third waveform; and during a second time interval, mixing the third and second waveforms by combining their respective sample values in proportions that vary during the second time interval.

25 According to a second aspect of the invention there is provided a system for generating a sound sequence comprising means for representing first, second and third waveforms by a plurality of sample values; means for mixing, during a first time interval, the first and second waveforms by combining their respective samples in proportions that vary during the first time interval;  
30 means for replacing the first waveform with the third waveform at the end of

the first time interval, when the proportion of the first waveform is zero; and means for mixing the third and second waveforms, during a second time interval, by combining their respective sample values in proportions that vary during the second time interval.

5

The invention further extends to a mobile phone and to an electronic toy which incorporates a system or which makes use of a method as previously mentioned.

10 According to another aspect of the present invention there is provided a method or a system in which a sound sequence is composed by repeated superimposition of discrete sample waveforms upon one another in defined proportions that vary during execution of the sequence.

15 The proportions, which may be proportions of the waveform amplitude, may be defined in terms of the proportions applicable at successive instants in the sequence. The proportions applicable in the intervals between successive ones of said instants may be determined in accordance with transition, for example for achieving a linear transition of amplitude, from one of those instants to the  
20 next. The intervals between the successive instants may be variable.

The method and system of the invention may be utilised in the context of a generative music (or other sound) system or other data processing system, for the generation of music (or other) sound sequence in that context, and in the  
25 economic low-bandwidth communication of data defining such sequences, by wireless or otherwise, from and to such systems. The invention is of especial significance in the transmission and rendering of audio content on networked devices.

30 The invention is of benefit in providing a low bandwidth solution for the

transmission of sounds, in that the sound sample waveforms can be described by a small set of values. This set is much smaller than the set required to render the sound itself (i.e. to a raw sample), hence the set of values can be more quickly transmitted than the raw sample. A further benefit is that it  
5 reduces the amount of computation required for creation of the desired complex waveforms, as it is in effect a hybrid approach by which a limited number of simpler, smaller pre-calculated waveforms can be mixed together in order to create the more-complex waveform.

10 A method and system for automated generation of sound sequences, and applications of such method and system, according to the invention will now be described, by way of example, with reference to the accompanying drawings, in which:

15 Figure 1 is a schematic representation of the system of the invention;

Figure 2 is illustrative of objects that are involved in a component of the system of Figure 1;

20 Figure 3 is a flow-chart showing process steps involved in control sequencing within the method and system of the invention;

Figure 4 is illustrative of operation of the method and system of the invention in relation to scale and harmony rules;

25

Figure 5 illustrates operation of the method and system of the invention in relation to the triggering of note sequences and their integration into a musical work as currently being composed and played;

30 Figure 6 shows in schematic form the way in which sample waveforms are

derived and communicated within the method and system of the present invention;

Figure 7 is illustrative of composition of a limited repetitive section of a sound sequence using the waveforms defined in Figure 6, in the method and system of the present invention; and

Figure 8 shows an alternative approach to that illustrated in Figure 7.

10 The method and system to be described are for automated generation of sound sequences and to integrate data presented or interpreted in a musical context for generating an output reflecting this integration. Operation is within the context of generation of musical works, audio, sounds and sound environments in real-time. More especially, the method and system function in the manner of a  
15 'generative music system' operating in real-time to enable user-interaction to be incorporated into the composition on-the-fly. The overall construction of the system is shown in Figure 1 and will now be described.

Referring to Figure 1, the system involves four high-level layers, namely, an  
20 applications layer I comprising software components 1 to 5, a layer II formed by an application programmer's interface (API) 6 for interfacing with a music engine SKME that is manifest in objects or components 7 to 14 of a layer III, and a hardware device layer IV comprising hardware components 15 to 19 that interact with the music engine SKME of layer III. Information flow between  
25 the software and hardware components of layers I to IV is represented in Figure 1 by arrow-heads on dotted-line interconnections, whereas arrow-heads on solid lines indicate an act of creation; for example, information in the composed-notes buffer 11 is used by the conductor 12 which is created by the soundscape 8.

The applications layer I determines the look, feel and physical instantiation of the music engine SKME. Users can interact with the music engine SKME through web applications 1, or through desktop computer applications 2 such as those marketed by the Applicants under their Registered Trade Mark KOAN as  
5 KOAN PRO and KOAN X; the music engine SKME may itself be such as marketed by the Applicants under the Registered Trade Mark KOAN. Interaction with the engine SKME may also be through applications on other diverse platforms 3 such as, for example through mobile telephones or electronic toys. All applications 1 to 3 ultimately communicate with the music  
10 engine SKME via the API 6 which protects the internals of the music engine SKME from the outside world and controls the way in which the applications can interact with it. Typically, the instructions sent to the API 6 from the applications 1 to 3 consist of commands that instruct the music engine SKME to carry out certain tasks, for example starting the composition and playback,  
15 and changing the settings of certain parameters (which may affect the way in which the music is composed/played). Depending on the needs of the individual applications, communication with the API 6 may be direct or via an intermediate API. In the present case communication to the API 6 is direct from the desktop computer applications 2, whereas it is via an intermediate  
20 browser plug-in API 4 and Java API 5 from applications 1 and 3 respectively.

The music engine SKME, which is held in memory within the system, comprises eight main components 7 to 14. Of these, SSFIO 7, which is for file input/output, holds a description of the parameters, rules and their settings used  
25 by algorithms within the engine, to compose. When the engine SKME is instructed via the API 6 to start composition/playback, a soundscape 8 is created in memory and this is responsible for creating a composer 10 (which runs in a background loop), conductor 12 and all the individual compositional objects 9 relating to the description of the piece as recorded in the SSFIO 7.  
30 The compositional objects are referred to by the composer 10 to decide what

notes to compose next. The composed notes are stored in a number of buffers 11 along with a time-stamp which specifies when they should be played. The conductor 12 keeps time, by receiving accurate time information from a timer device 19 of level IV. When the current time exceeds the time-stamp of notes  
5 in the buffers 11, the relevant notes are removed from the buffers 11 and the information they contain (such as concerning pitch, amplitude, play time, the instrument to be used, etc.) is passed to the appropriate rendering objects 13. The rendering objects 13 determine *how* to play this information, in particular whether via a MIDI output device 17, or as an audio sample via an audio-out  
10 device 18, or via a synthesiser engine 14 which generates complex wave-forms for audio output directly, adding effects as needed.

The hardware devices layer IV includes in addition to the devices 17 to 19, a file system 15 that stores complete descriptions of rules and parameters used  
15 for individual compose/playback sessions in the system; each of these descriptions is stored as an 'SSfile', and many of these files may be stored by the file system 15. In addition, a MIDI in device 16 is included in layer IV to allow note and other musical-event information triggered by an external hardware object (such as a musical keyboard) to be passed into the music  
20 engine SKME and influence the composition in progress.

The system can be described as having essentially two operative states, one, a 'dynamic' state, in which it is composing and the other, a 'static' state, in which it is not composing. In the static state the system allows modification of the  
25 rules that are used by the algorithms to later compose and play music, and keeps a record encapsulated in the SSFIO component 7, of various objects that are pertinent to the description of how the system may compose musical works. The system is also operative in the dynamic state to keeps records of extra objects which hold information pertinent to the real-time composition and  
30 generation of these works. Many of these objects (the compositional objects 9



for example) are actual instantiations in memory of the descriptions contained in the SSFIO 7. Modification of the descriptions in the SSFIO 7 via the API layer II during the dynamic state, results in those modifications being passed down to the compositional objects 9 so that the real-time composition changes accordingly.

Figure 2 shows a breakdown of the SSFIO component 7 into its constituent component objects which exist when the system is in its static and dynamic states; the system creates real-time versions of these objects when composing and playing. In this respect, SSfiles 20 stored each provide information as to 'SSObject(s)' 21 representing the different *types* of object that can be present in the description of a work; these objects may, for example, relate to piece, voice, scale rule, harmony rule, rhythm rule. Each of these objects has a list of 'SSFparameters' 22 that describe it; for example, they may relate to tempo, instrument and scale root. When an SSfile 20 is loaded into the music engine SKME, actual instances of these objects 21 and their parameters 22 are created giving rise to 'SSFObjectInstance' 23 and 'SSFParameterInstance' 24 as illustrated in Figure 2.

Referring again to Figure 1, the user interacts with the system through applications 1 to 3 utilising the services of the API 6. The API 6 allows a number of functions to be effected such as 'start composing and playing', 'change the rules used in the composition', 'change the parameters that control how the piece is played' including the configuration of effects etc. One of the important advantages of the described method and system is the ability to trigger generative pattern sequences in response to external events. The triggering of a generative pattern sequence has a range of possible outcomes that are defined by the pattern sequence itself. In the event that a generative pattern sequence is already in operation when another trigger event is received, the currently operational sequence is ended and the new one scheduled to start

at the nearest availability.

Generative pattern sequences allow a variety of musical seed phrases of any length to be used in a piece, around which the music engine SKME can  
5 compose in real time as illustrated in Figure 3. More particularly, the generative pattern sequence contains a collection of one or more note-control sub-patterns with or without one or more additional sequence-control sub-patterns. Three types of note-control sub-patterns can be created, namely: 'rhythm' note-control sub-pattern containing note duration information, but not  
10 assigning specific frequencies to use for each note; 'frequency and rhythm' note-control sub-pattern containing both note duration and some guidance to the generative music engine SKME as to the frequency to use for each note; and 'forced frequency' note-control sub-pattern containing note duration, temporal positioning and explicit frequency information to use for each note.  
15 Sequence-control sub-patterns, on the other hand, can be used to specify the sequence in which the note-control sub-patterns are played, and each note-control sub-pattern may also specify ranges of velocities and other musical information to be used in playing each note. The music engine SKME allows the use of multiple sub-patterns in any generative pattern sequence.

20

Referring to Figure 3, the step 30 of triggering the generative pattern sequence acts through step 31 to determine whether there are any other sequence-control sub-patterns operative. If not, a note-control sub-pattern is chosen at random in  
25 step 32 from a defined set; each note-control sub-pattern of this set may be assigned a value that determines its relative probability of being chosen. Once it is determined in step 33 that the selected note-control sub-pattern is finished, another (or the same) note-control sub-pattern is selected similarly from the set. The generative pattern sequence continues to play in this manner until instructed otherwise.

30

If the result of step 31 indicates that there is one or more sequence-control sub-patterns operative, then any sequence-control sub-pattern is chosen at random in step 34 from the defined set; each sequence-control sub-pattern may be assigned a value that determines its relative probability of being chosen. Once  
5 a sequence-control sub-pattern has been selected in step 34, it is consulted to determine in step 35 a sequence of one or more note-control sub-patterns to play. As each note-control sub-pattern comes to an end, step 36 prompts a decision in step 37 as to whether each and every specified note-control sub-pattern of the operative sequence has played for the appropriate number of  
10 times. If the answer is NO, then the next note-control sub-pattern is brought into operation through step 35, whereas if the answer is YES another, or the same, sequence-control sub-pattern is selected through repetition of step 34. As before, the generative pattern sequence continues to play in this manner until instructed otherwise.

15

Each sequence-control sub-pattern defines the note-control sub-pattern(s) to be selected in an ordered list, where each entry in the list is given a combination of: (a) a specific note-control sub-pattern to play, or a range of note-control sub-patterns from which the one to play is chosen according to a relative  
20 probability weighting; and (b) a value which defines the number of times to repeat the selected note-control sub-pattern, before the next sequence-control sub-pattern is selected. The number of repetitions may be defined as a fixed value (e.g. 1), as a range of values (e.g. repeat between 2 and 5 times), or as a special value indicating that the specified note-control sub-pattern should be  
25 repeated continuously.

Depending upon the note-control sub-pattern operational at any moment after a generative pattern sequence is triggered, various rules internal to the music engine SKME may be used to determine the exact pitch, duration and temporal  
30 position of the notes to be played. For example, if a 'rhythm' note-control sub-

pattern is in operation at a particular point in the generative pattern sequence, then the scale rule, harmony rule and next-note rule within the music engine SKME for that 'triggered voice' will be consulted to obtain the exact notes. Alternatively, if the 'forced frequency' note-control sub-pattern is operational,  
5 no internal rules need be consulted since all the note information is already specified. Furthermore, for the case of 'frequency and rhythm', the music engine SKME combines the given frequency offset information with its rules and other critical information such as the root of the current scale and range of available pitch values for the voice in question.

10

The rules and other parameters affecting composition (e.g. tempo) within the music engine SKME are defined in memory, specifically within the SSFIO 7, and its real-time instantiation of the compositional objects 9. Use of rules and parameters within the music engine SKME form part of the continual  
15 compositional process for other voice objects within the system. Figure 4 illustrates this more general process based on examples of scale and harmony rules shown at (1) and (2) respectively.

Referring to Figure 4, the scale rule is illustrated at (1) with shaded blocks  
20 indicating a non-zero probability of choosing that interval offset from a designated scale root note. The larger the shaded block, the greater the probability of the system choosing that offset. Thus, for this example, the octave Ove, major third M3 and fifth 5 are the most likely choices, followed by M2, 4, M6 and M7; the rest will never be chosen. Sequences that may be  
25 generated by the system from this are shown below the blocks, and in this respect the octave has been chosen most often followed by the major third and the fifth. With the scale root set in the system as C, the resulting sequence of notes output from the system in this example are C,E,C,D,G,A,E,D,C,G,E,B,C,F, as illustrated at (1) of Figure 4.

The harmony rule defines how the system may choose the pitches of notes when other notes are playing, that is to say, how those pitches should harmonise together. In the example illustrated at (2) of Figure 4, only the octave and major second are indicated (by shading) to be selected. This means  
5 that when the pitch for a voice is chosen, it must be either the same pitch as, or a major second from, all other notes currently being played.

For the purpose of further explanation, consideration will be given to the example represented at (3) of Figure 4 involving three voice objects V1-V3.  
10 The rhythm rules applicable to the voice objects V1-V3 in this example, give rise to a generated sequence of notes as follows: voice V1 starts playing a note, then voice V2 starts playing a note, then voice V3 starts playing a note, and then after all notes have ended, voice V2 starts playing another note, followed by voice V1 and then voice V3. With this scenario, the note from voice V2  
15 must harmonise with that of voice V1 and the voice V3 note must harmonise with that of voice V2. If in these circumstances the voice V1 is, as illustrated by bold hatching, chosen with a pitch offset of a fifth from the scale root, the pitch for voice V2 must either be the same as (Ove) or a major second above (M2) the fifth. In the case illustrated, it is chosen to be the same, and so the  
20 fifth is chosen too. When voice V3 starts playing it must harmonise with both voices V1 and V2, so the pitch chosen must be the same as, or a major second above that of voices V1 and V2. As illustrated, the system chooses voice V3 to be a major second above, therefore giving pitch offset M6 from the scale root.

25 After voice V3 all notes end, and the next note begins, as illustrated at (4) of Figure 4 with voice V2. This next note by voice V2 is governed by the next-note rule used by voice V2, and the last note played by voice V2. According to this rule, the system chooses pitch offset M2 for voice V2, and then harmonises voices V3 and V1 with it by choice of a major second for both of them. With  
30 the scale root set in the system to C, the entire generated sequence accordingly

follows that indicated at (5) of Figure 4, where 'S' denotes a note starting and 'E' a note ending.

Thus, when sequences are generated in response to an external trigger, the actual pitches and harmonisation of that sequence is determined by the composer 10 using several items of information, namely: (a) the note-control sub-pattern operational at that moment; (b) the scale, rhythm, harmony and next-note rules depending upon the type of the note-control subsequence; and (c) any piece-level rules which take into account the behaviour of other voices within the piece.

When the music engine SKME is in dynamic (i.e. composing and playing) mode, it typically contains a number of voice compositional objects 9. The composer 12 composes a sequence of notes for each of these and makes sure they obey the various rules. The process involved is illustrated in the flow diagram of Figure 5.

Referring to Figure 5, the music engine SKME responds to an external trigger applied at step 51, and the API 6 through step 52 instructs a voice 1 in step 53 to register that it must start a sequence. Voice 1 and the voices 2 to N in step 54, have their own rules, and the composer 10 ensures that the relevant rules are obeyed when utilising any of the voices 1 to N. More particularly, the composer 10 responds in step 55 to the instruction of step 53 for voice 1 to start a sequence, by starting the generative pattern sequence sub-system of Figure 3. This sends note-control sub-sequences to the trigger voice (voice 1 in this example), but the composer 10 makes sure the resulting notes harmonise with the other voices in the piece. The outcome via the conductor 12 in step 56 is played in step 57.

The generative pattern sequence triggered will play forever, or until the system

is instructed otherwise. If a sequence control sub-pattern is used to define a generative pattern sequence such that the final note control sub-pattern is one which plays silence (rest notes) in an infinite loop, then when this pattern sequence is selected, the voice will become effectively 'inactive' until another trigger is detected. Further triggering events for the same generative pattern sequence may sound different as the process is generative, or since the rules in use by the piece or the scale of the trigger voice, its harmony or next note rules may have changed (either via interaction through the API 6 or via internal music engine SKME changes).

10

The sounds used to 'render' each note, whether from triggered sequences or generative voices may be played either through the MIDI sounds or the samples of the rendering objects 13, or via software of the synthesiser engine 14 which may add digital signal processing effects such as, for example, filter sweeps and reverberation chorus. The entire process can be used to generate musical event information that is then fed into, and may thus control, other processing units within the system such as synthesiser related units allowing the triggering of generative sound effects. Voices can also be added which make use of the software synthesiser engine 14 to generate non note-based effects such as sound washes and ambient environmental sounds, such as chimes, wind and other organic sounds.

20

The present invention takes advantage of the ability of the generative music system to integrate new structure into an existing musical content. In this respect, a number of values describing the harmonics present in a waveform and how those harmonics change over time and in relative amplitude are used as input to a set of algorithms or equations in a 'virtual wave generator module' that can create an output waveform from this data. The virtual wave generator module is resident in the synthesiser engine 14 of the generative music system.

25

30

The waveform description can be saved to a file or a text string and then transmitted to another application, for example through e-mail or a binary file or through the communications protocol known as BLUETOOTH, and the virtual wave generator module in that application may render the audio from that set of waveform descriptions. In this way a rich waveform can be created containing many harmonics which change over time, without explicitly having to sum all the required harmonics for each sample point. By contrast, summation is of sample points from a small number of pre-calculated samples each of which contains many harmonics. This results in greater computational efficiency and economy in communication, without losing richness in the resulting sound.

The operation is illustrated in Figures 6 and 7 and will now be described.

Referring to Figure 6, several small waveforms 61 to 63 are generated, each of one wavelength (the fundamental wavelength) in duration; this can be done quickly and economically since only the one wavelength is needed in each case. The three waveforms 61 to 63 (there may be more than three) are stored as 'samples', each being defined in this regard by the relative proportions of the amplitudes of the harmonics contained within it. In this particular example, the definition is in terms of the relative amplitudes of the first to sixth harmonics; more or fewer harmonics may be involved. Sample waveform 61 is in this context accordingly defined as shown by the string 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, indicating that it comprises just the first and second harmonics with amplitudes in the proportions 1.0:0.3, sample waveform 62 is defined by the string 0.0, 0.0, 1.0, 0.4, 0.2, 0.0, indicating that it comprises just the third, fourth and fifth harmonics with amplitudes in the proportions 1.0:0.4:0.2, whereas sample waveform 63 is defined by the string 0.0, 0.0, 0.0, 0.0, 0.5, 0.3 indicating that it comprises just the fifth and sixth harmonics with amplitudes in the proportions 0.5:0.3.



The waveforms need not be sinusoidal, but could instead be internally-generated waveforms of other shapes (e.g. sawtooth etc). Alternatively, each waveform may be a downloaded micro-sample.

5

The sample waveforms 61 to 63 are summed together repetitively and played in a loop with their relative contributions to the output varying with time. Typically, each waveform repeats about every 2 ms. This is illustrated in Figure 7 in respect of a section of the rendering of the created sequence.

10

Referring to Figure 7, a number of serially linked data elements 71 to 75 are defined in memory. Each of the elements 71 to 75 specifies the relative contributions that the waveforms 61 to 63 make to the total played output at a specific instant of the output sequence. In addition, each element 71 to 75 includes definition of the duration for which the relative contributions of the waveforms 61 to 63 are to transition to those of the next element of the series.

Thus in Figure 7, the five elements 71 to 75 become operative in turn to provide an output music (or other sound) sequence that comprising five intervals 76 to 80, the first of which, interval 76, begins when element 71 becomes effective. The duration of interval 76 is defined in element 71 as 1 second, and the output rendering at the beginning of this interval 76 is created by summing or superimposition upon one another of 100% of waveform 61, 50% of waveform 62 and 0% of waveform 63. As the creation of the output progresses through the interval 76 of 1 second duration, so the component waveforms 61 to 63 transition linearly in their proportions to those of the next element 72. Accordingly, at the start of the next interval 77 the output is made up of 0% of waveform 61, 100% of waveform 62 and 10% of waveform 63. The process continues for the next 0.5 second from that waveform composition defined by element 72, to that of element 73, so that at instant t mid-way

through the interval 77 the output is composed of 20% of waveform 61 (mid-way between 0% and 40%), 50% of waveform 62 (mid-way between 100% and 0%) and 55% of waveform 63 (mid-way between 10% and 100%).

- 5 Play continues in the same way through intervals 78 to 80 of the sequence, each interval 78 to 80 starting with the relative proportions of the waveforms 61 to 63, defined by the respective element 73 to 75. The sequence is looped so as to be repeated, the proportion of waveforms 61 to 63 transitioning during interval 80 from those of element 75 to those of element 71. The resulting sound is that
- 10 of many harmonics changing and blending quickly or slowly over time, and in that regard it is possible to impose variation on the data within any individual element so as to effect change in the output sound, by for example, extending the duration-time stored for the element and/or varying individually or collectively the definitions of amplitude of the waveforms 61 to 63, defined in
- 15 it. Pitch of the sound produced may be varied by extending the waveform duration, and indeed the composition in terms of the relative proportions of the fundamental and its harmonics, of each waveform 61 to 63 may be varied by simple variation of numerical/textural parameters.
- 20 The method of the invention in which the output is created from elements defining the contributions from a limited number (three in the above example) of looping waveforms has significant advantage over the heavy processing involved in creating the same sound output from summing harmonics. For example, the transmission of data necessary from the applications layer I
- 25 (Figure 1) to bring about creation of the music or other sound sequence by the music engine SKME, can be limited to text or other definition of the limited number of waveforms, and then followed simply by data defining the elements for controlling composition in terms of proportions of those waveforms and duration-times. Waveform definitions may in any event be stored in the music
- 30 engine SKME for selection, reducing further the bandwidth required for the

information that needs to be sent.

It will be appreciated that the present invention enables a small set of values to be used to define one or more waveform samples for the provision of sound sequences, not only of music but of speech and other sounds, under the control of small data elements defining how those waveforms are repeated and combined into an integrated composition. Although, as in the above example described with reference to Figure 7, the waveforms 61 to 63 are combined directly together within each interval 76 to 80, the output may instead be produce by first deriving separate sequences each composed of a respective waveform 61 to 63 in the appropriately changing proportions defined by the elements 71 to 75, and then combining these sequences together to provide an integrated output.

In the example of Figure 7, the chosen waveforms W1, W2 and W3 remain the same for the entire length of the output. That is, however, not essential, and any of the waveforms could be changed for a different one at any point during the sequence when that waveform contributes zero percent to the output. For example, in Figure 7, waveform 1 could be changed for another waveform at the end of the interval 76 or at the end of the interval 78. It would also be possible to change the interval lengths 76 to 80 on the fly, rather than defining the lengths in advance. The interval lengths could, for example, be determined by the software itself, without explicit user input.

Not all of the waveforms necessarily need to be sampled, as shown in Figure 6. It would also be possible to mix in a certain amount of white noise along with the musical waveforms, the proportion varying with time in the same way that the waveform proportions vary with time.

Figure 8 shows a development of the approach in which the first sample is

played once only, with subsequent samples being repeated. As shown in the Figure, once the output finishes, at 81, control returns as shown by the dotted line 82 to an internal point 83. Thus, the section 76 is played once only, while the sections 77 to 81 are continually repeated. This allows for the composer to  
5 introduce additional complexity into the section 76 allowing, for example, the creation of a "plucked" sound (or a distinct attack) followed by a resonance effect.

Other loops are of course possible. For example, it may on occasion be  
10 desirable to play the first section once only, repeat the central section N times, and play the final section once only. More generally, loops can start and finish anywhere within the overall sequence, and may be repeated any desired number of times. An output sequence may also make use of more than one loop, with the different loops covering different intervals, and possibly defining differing  
15 numbers of repeats.

The approach described above may be used to create an ambient or background sound with depth, richness and timbre. Generative music, created as described in Figures 1 to 5, may be played over the top of this.

20

The approach has particular application to embedded devices, allowing users to generate and to manipulate on the fly a rich, deep, sound-structure without the need to download individual samples across a network. Particular applications include mobile telephones, where there is a need to create rich and distinctive  
25 ring tones, generated internally, and also electronic toys.

**CLAIMS:**

1. A method of generating a sound sequence comprising representing first, second and third waveforms by a plurality of sample values; during a first  
5 time interval, mixing the first and second waveforms by combining their respective samples in proportions that vary during the first time interval; at the end of the first interval, when the proportion of the first waveform is zero, replacing the first waveform with the third waveform; and during a second time interval, mixing the third and second waveforms by combining their  
10 respective sample values in proportions that vary during the second time interval.
2. A method as claimed in claim 1 in which the length of the second time interval is changed during the period of the first time interval.  
15
3. A method as claimed in claim 1 or claim 2 in which during the first or the second time interval a variable amount of white noise is combined with the waveform sample values.
- 20 4. A method as claimed in any one of the preceding claims including locally generating the sample values.
5. A method as claimed in any one of claims 1 to 3 including receiving the sample values across a network.  
25
6. A method as claimed in any one of the preceding claims in which more than two waveforms are combined during the first or the second time intervals, or both.
- 30 7. A method as claimed in any one of the preceding claims in which the

sound sequence generated is a ring tone for a mobile phone.

8. A method as claimed in claims 1 to 7 in which the sound sequence is generated by an electronic toy.

5

9. A system for generating a sound sequence comprising means for representing first, second and third waveforms by a plurality of sample values; means for mixing, during a first time interval, the first and second waveforms by combining their respective samples in proportions that vary during the first  
10 time interval; means for replacing the first waveform with the third waveform at the end of the first time interval, when the proportion of the first waveform is zero; and means for mixing the third and second waveforms, during a second time interval, by combining their respective sample values in proportions that vary during the second time interval.

15

10. A system as claimed in claim 9 including means for locally generating the sample values.

11. A system as claimed in claim 9 including means for downloading  
20 remotely-generated sample values across a network.

12. A mobile phone incorporating a system as claimed in claim 9.

13. A mobile phone as claimed in claim 12 in which the sound sequence  
25 generated is a ring tone.

14. An electronic toy incorporating a system as claimed in claim 9.

1 / 7

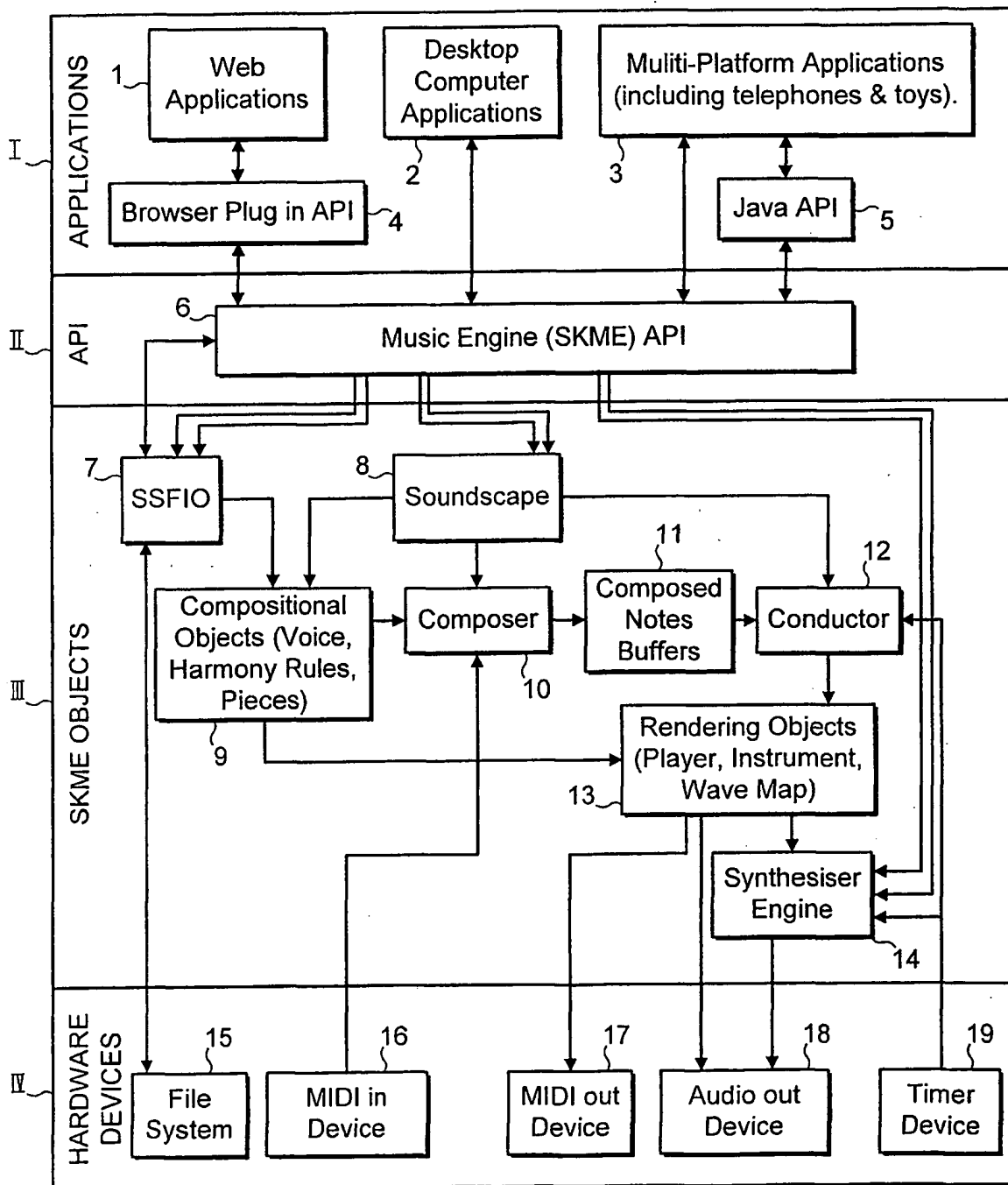


FIG. 1

2 / 7

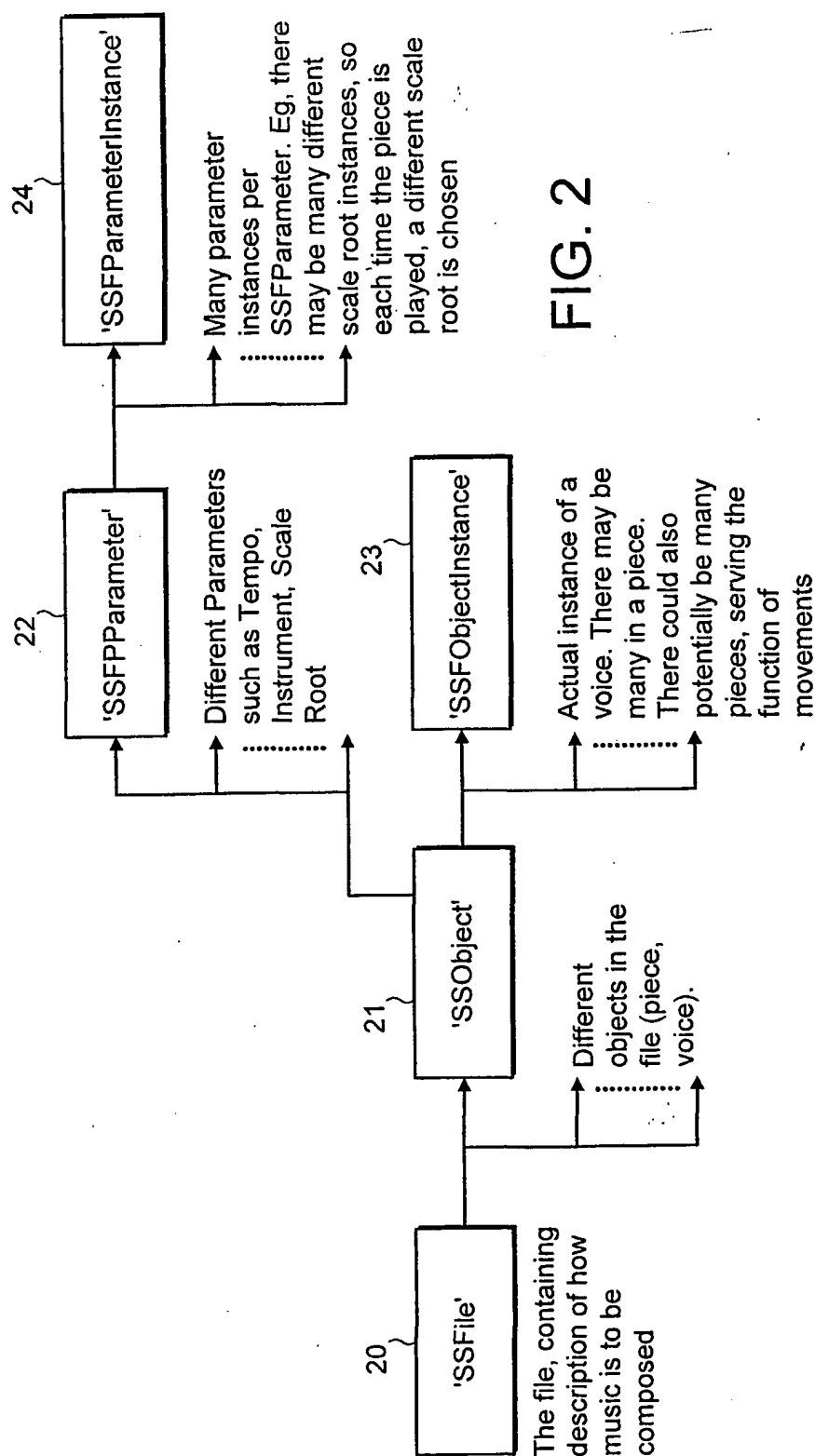


FIG. 2



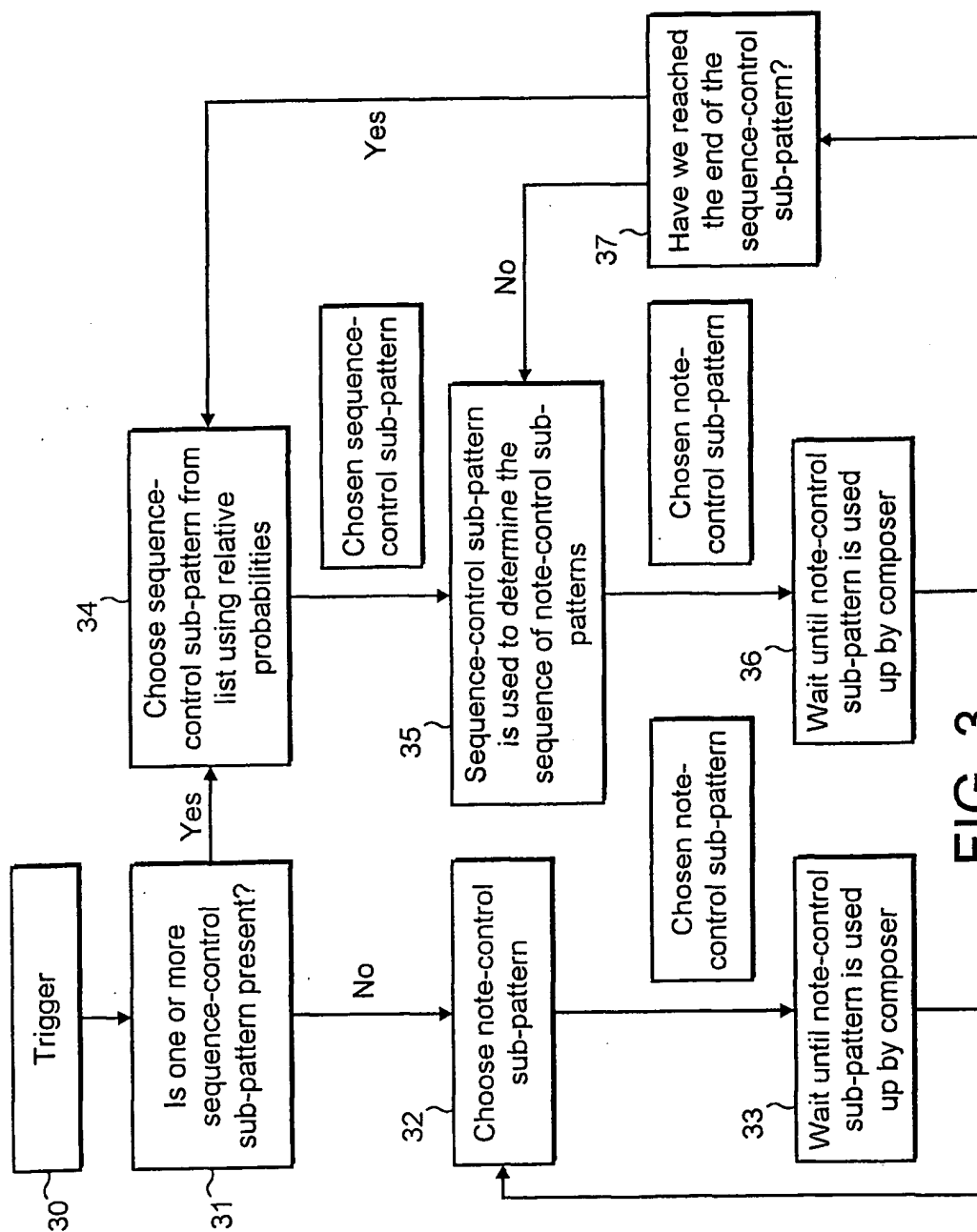
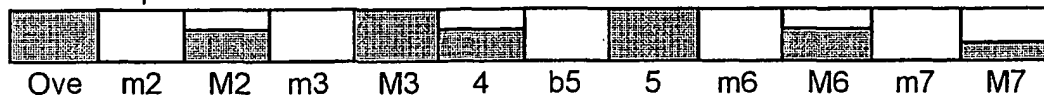


FIG. 3

4 / 7

## 1 An example Scale Rule:



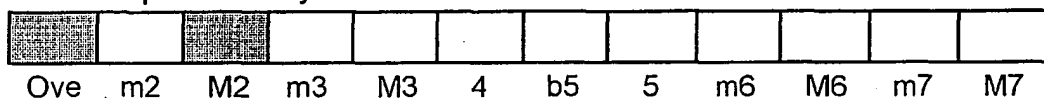
Relative to a scale root, a sequence of notes chosen from this scale rule might be:

Ove, M3, Ove, M2, 5, M6, M3, M2, Ove, 5, M3, M7, Ove, 4

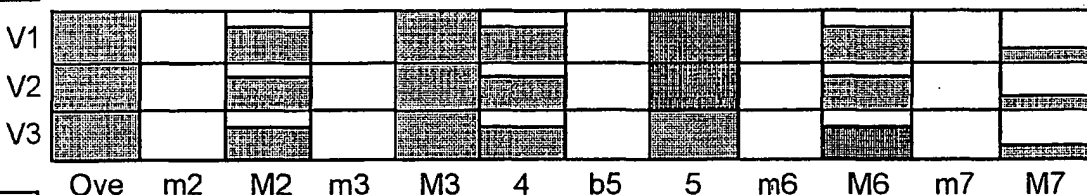
With the scale root set at C, this sequence becomes:

C, E, C, D, G, A, E, D, C, G, E, B, C, F

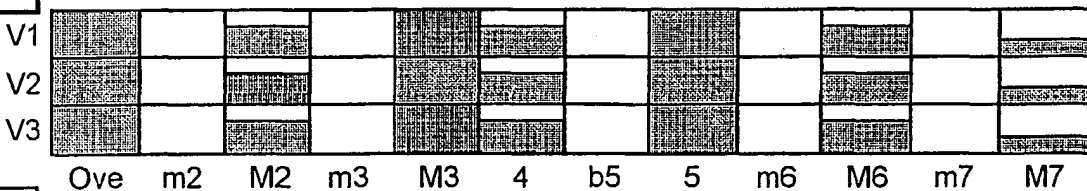
## 2 An example Harmony Rule:



## 3



## 4



## 5

Time Sequence:

S:V1:G, S:V2:G, S:V3:A,

E:V1, E:V2, E:V3

S:V2:D, S:V3:E, S:V1:E

E:V2, E:V3, E:V1

FIG. 4

5 / 7

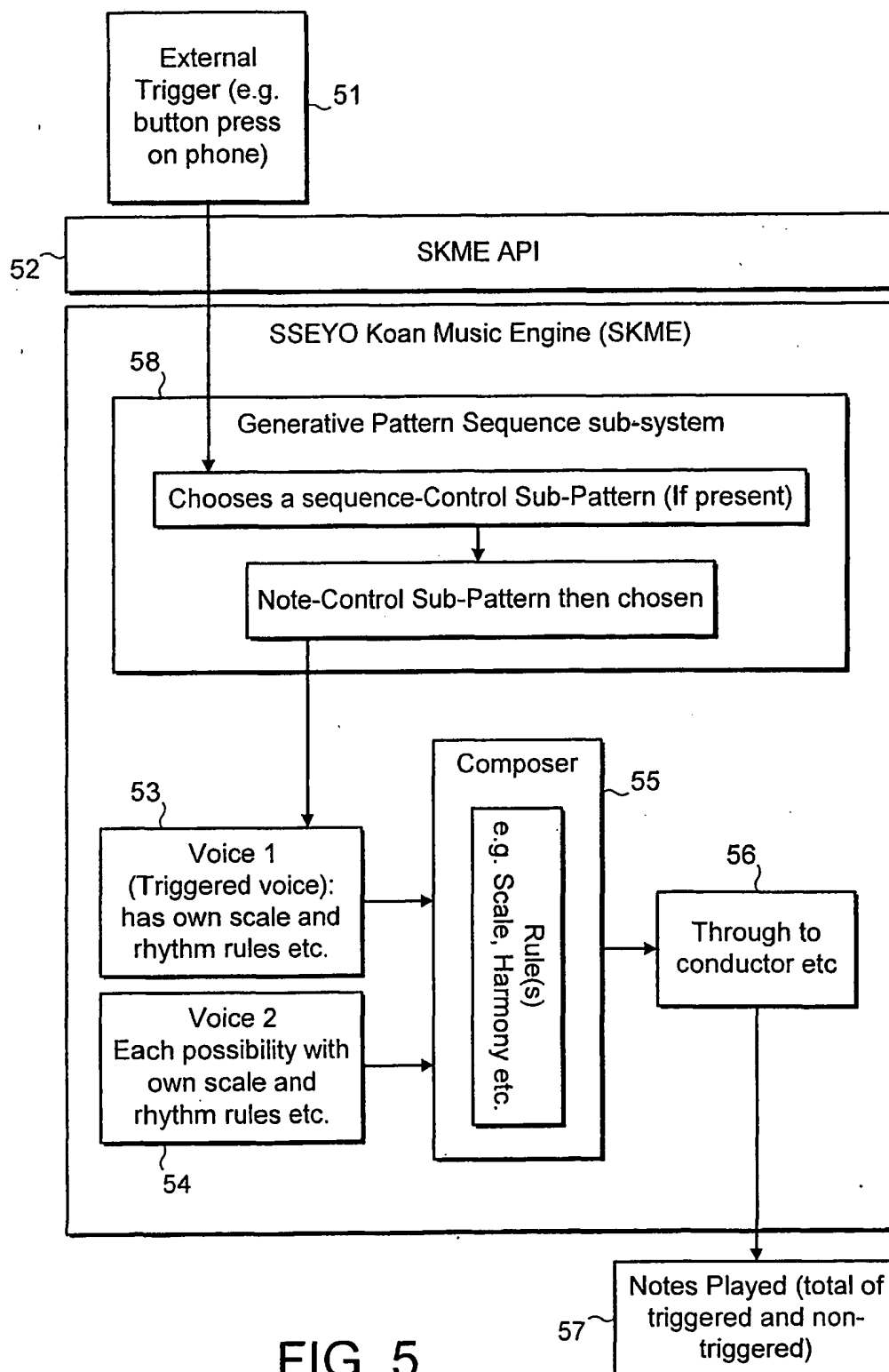
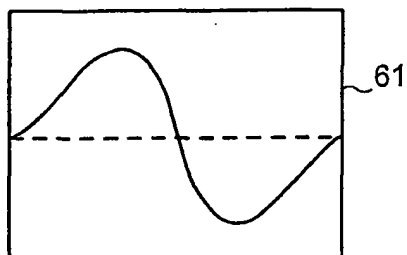
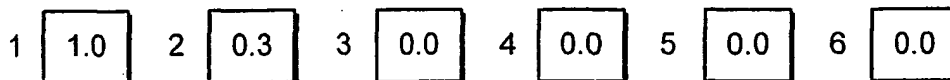


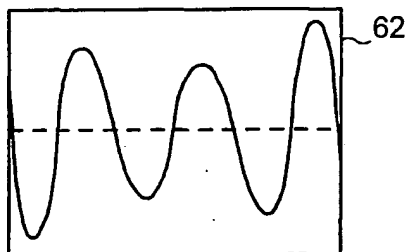
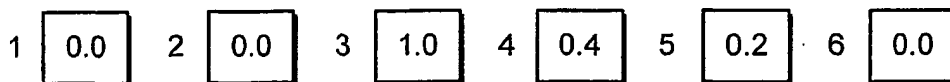
FIG. 5

6 / 7

Wave No 1



Wave No 2



Wave No 3

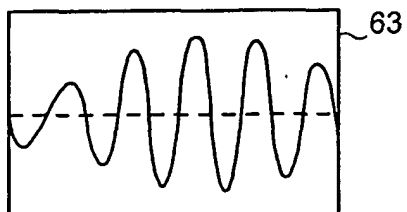
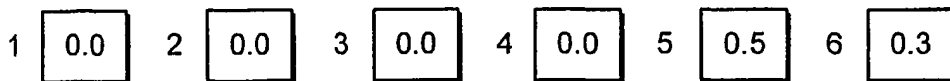


FIG. 6

7 / 7

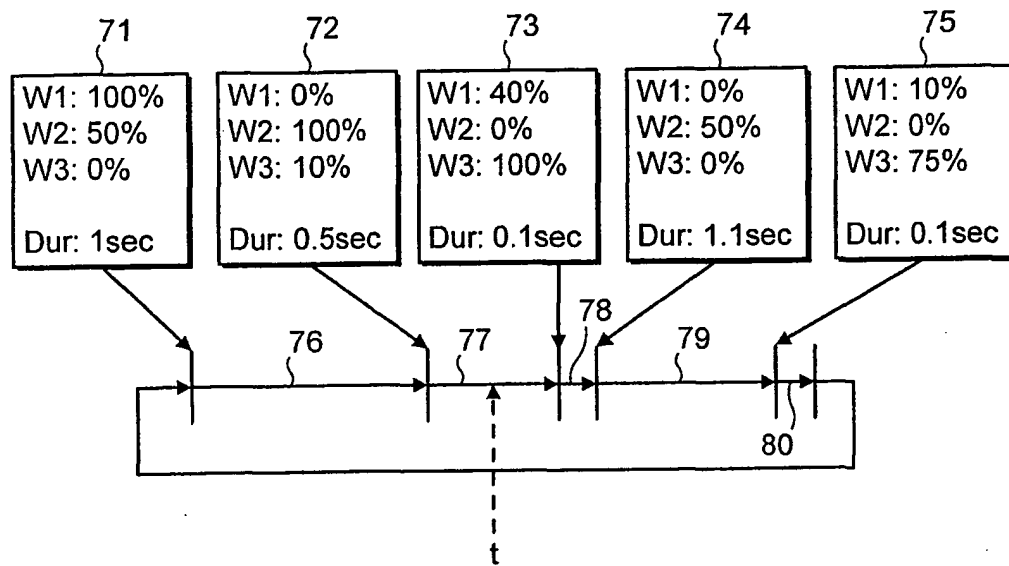


FIG. 7

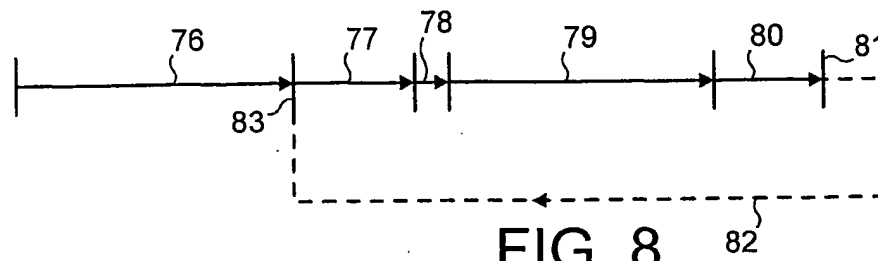


FIG. 8